IN THE SPECIFICATION:

Please add the following paragraph before "FIELD OF THE INVENTION" on page 1, line 1:

--CROSS REFERENCE TO RELATED APPLICATIONS

This application is a continuation of application Serial No. 09/608,959 filed June 30, 2000, the contents of which is incorporated herein in its entirety by reference thereto.--

Please amend the following specification paragraphs.

Please amend the second full paragraph starting on page 4, line 16 and bridging to page 6, line 1 as follows:

"FIG. 1 is a high-level logic block diagram illustrating an error detection apparatus being used to protect pre-validated region identifications (RIDs) in the translation look-aside buffer (TLB), in accordance with an embodiment of the present invention. In FIG. 1, in accordance with an embodiment of the present invention, a simple 1-bit parity scheme is used by an error detection apparatus 100 to protect a pre-validated RID array 145 in a TLB 140. The error detection apparatus 100 includes an error detection state machine (EDSM) 110, a shift register 120, an XOR gate 125 coupled to the shift register 120 and a feedback channel 131, a result latch 130 coupled to the XOR gate 125 and the EDSM 110, and an XOR gate 135 coupled to the result latch 130 and the EDSM 110. The TLB 140 includes the RID array 145, a RID parity bit array 146, a protection key register array 150, a protection key register parity bit array 151, an enable/disable bit 155, a virtual address 160 and a physical address 165. In accordance with an embodiment of the present invention, the EDSM 110 periodically reads out the content of an entry (a row) in the RID array 145, which includes the parity bit for the row, stores the read-out content in the shift register 120 and computes the parity bit for the read-out

entry in a serial fashion by the XOR gate 125. In, general, at the end of the parity computation step as indicated by counter ~~160~~ 118, the computed parity bit in the result latch 130 will be stored in the parity and valid bits register ~~150~~ 119 if the valid bit is not set. If the valid bit in the parity and valid bits register ~~150~~ 119 is set, the parity of the result latch 130 will be compared with the parity bit in the parity and valid bits register ~~150~~ 119. A mismatch results in a Machine Check Abort (MCA) signal being out. This MCA will cause the processor to transfer execution control to the firmware error handler to take proper error recovery action. The EDSM 110 uses the existing read port into the region ID or protection key arrays. When computing the parity bit, the content of the RID array 145 is read into the shift register 120. The content of the shift register 120 is then shifted out 1 bit at a time and the output of the shift register 120 is fed into the XOR gate 125. At the end of the shift operation as indicated by the EDSM 110, the polarity of the result latch 130, which indicates whether the entry will have a parity error, is output to the XOR gate 135 and fed back to the XOR gate 125. The end-of-shift signal that is output from the EDSM 110 is also received by the XOR gate 135.

Please amend the second full paragraph on page 6, lines 2 through 19 as follows:

In FIG. 1, in accordance with an embodiment of the present invention, the EDSM 110 includes a timer 112, a next pointer 114 and a move to RID indicator 116. The timer 112 is fired periodically to initiate a read of a row from the RID array 145 into the shift register 120. The counter ~~160~~ 118 counts the number of shifts needed to compute the proper parity bit. If the valid bit is not set, then the parity and valid bits register ~~150~~ 119 stores the result of the parity bit computation; otherwise, the parity and valid bits register ~~150~~ 119 stores the parity bit used to compare with the parity bit in the result latch 130. The EDSM 110 also remembers which entry to read out via the next pointer 114 logic, which stores a pointer that is used to determine which row in the RID array 145 is to be read-out, for example, the pointer can be used to indicate the actual row to

be read-out or the last row that was read out. The move to RID 116 logic monitors any incoming move to RID operation and will also invalidate the valid bit of the parity and valid bits register ~~150~~ 119. The EDSM 110 will not do a read if there is an incoming RID operation. Note that, in other embodiments of the present invention, the XOR gate 125 can be replaced by other logic to implement other arbitrary and more complicated protection schemes, such as, for example, checksumming the entire contents of the RID array 145. In such embodiments, for example, a pre-validated checksum value can be associated with the RID array 145, either in place of or in conjunction with the parity bit array 146, the entire contents of the RID array 145 can be read out, and a new checksum value can be computed by the checksum logic.

Please amend the second full paragraph starting on page 8, line 13 and bridging to page 9, line 2 as follows:

In FIG. 2, the MSR EDSM 210 uses an existing read port into the CRAB bus read logic 225 to request the next entry to be read out. When computing the parity bit, the content of an MSR is read into the shift register 120. The content of the shift register 120 is then shifted out 1 bit at a time and the output of the shift register 120 is fed into the XOR gate 125. Specifically, on the final shift, as indicated by the counter ~~160~~ 118, the parity bit in the result latch 130 will be stored into the parity and valid bits register ~~150~~ 119 if the valid bit is not set. If the valid bit is set, the computed parity bit will be compared with that in the parity and valid bits register ~~150~~ 119. At the end of the shift operation as indicated by the MSR EDSM 210, the polarity of the result latch 130, which indicates whether the entry will have a parity error, is output to the XOR gate 135 and fed back to the XOR gate 125. The end-of-shift signal that is output from the MSR EDSM 210 is also received by the XOR gate 135.

Please amend the first full paragraph on page 9, lines 3 through 14 as follows:

In FIG. 2, in accordance with an embodiment of the present invention, the MSR

EDSM 210 includes a timer 112, a next pointer 114 and a move to MSR indicator 216. The MSR EDSM 210, in FIG. 2, operates in the same manner as the EDSM 110 of FIG. 1. The timer 112 is fired periodically to initiate a read of an MSR 220a-220f into the shift register 120 to check if the read-out MSR is still valid. The MSR EDSM 210 also remembers which entry to read out via the next pointer logic 114, which stores a pointer that is used to determine which of the MSRs 220a-220f is to be read-out, for example, the pointer can indicate the actual row to be read-out or the last row that was read out. The move to MSR 216 logic monitors any incoming move to MSR operation. The MSR EDSM 210 will not do a read if there is an incoming MSR operation. Note that the XOR gate 125 can be replaced by other logic to implement other arbitrary and more complicated protection schemes such as, for example, checksumming the entire content of the MSR array, which may be stored in a separate checksum component 230.~~.~~

Please amend the second full paragraph starting on page 9, line 15 and bridging to page 11, line 2 as follows:

FIG. 3 is a detailed functional flow diagram of the operation for the error detection apparatus' detailed in FIGs. 1 and 2, in accordance with an embodiment of the present invention. In FIG. 3, in block 310, depending on what the EDSM is protecting, the contents of a row in the RID 145 and the associated parity bit from parity bit column 146 or one of the MSRs are periodically read out and, then, in block 320, the contents are stored in the shift register 120. In block 330, a parity bit or checksum for the contents stored in the shift register is calculated. This calculation is accomplished by shifting out the contents of the shift register 120 1 bit at a time and feeding each shifted-out bit into the XOR gate 125 along with a bit signal on feedback line 131. In block 340, if the valid bit is not set, then the parity and valid bits register ~~150~~ 119 stores the result of the parity bit computation; otherwise, the parity and valid bits register ~~150~~ 119 stores the parity bit that is to be used to compare with the computed parity bit in the result latch 130. If the

valid bit is set, then, the computed parity bit is compared to the stored parity bit in the parity and valid bits register ~~150~~ 119, to determine if they are equal, which would indicate that the contents in the RID 145 or MSR are valid. The XOR gate 125 outputs a polarity as a result of this comparison of bits and sends the polarity to the result latch. If the stored and computed parity bit or checksum in the XOR gate 125 then the polarity goes and the associated parity bit low and, if the stored and output values do not match, then the polarity goes high. In block 350, if the stored and output values do not match, then a machine check abort (MCA) signal is output. The MCA causes the processor error recovery firmware to be activated and to take the necessary error correction action. Outputting the MCA is performed by the XOR gate 135 when the polarity of only one of either the stored parity bit received from the parity and valid bits register ~~150~~ 119 and the final output from the result latch are high.

In accordance with an embodiment of the present invention, when the processor is reset, the contents of some of the MSRs 220a-220f may not be valid and the processor firmware is responsible for initializing the MSRs 220a-220f. Generally, after the MSRs are initialized, the processor firmware computes the effective parity and then writes it into the parity and valid bits register ~~150~~ 119 inside the MSR EDSM 210 and then starts the MSR EDSM 210. The MSR EDSM 210 will only compare the computed parity bit (or checksum) when the valid bit is set. The values in the MSRs 220a-220f may also be changed. In this case, PAL disables the MSRs 220a-220f and then updates the MSR. The parity bit or checksum is then re-computed and updated in the MSR EDSM 210.